ALP4490/ALP4491

Japanese Patent Laid-Open No.59-62968

(12) Japanese Patent Laid-Open (A) No. 59-62968

(21) Application No. 57-170841

(22) Date of filing: October 1, 1982

(72) Inventor: Obayashi Masanao et al.

(71) Applicant: Hitachi, Ltd

5-1, Marunouchi 1-chome, Chiyoda-ku, Tokyo

[Title of the Document]

Specification

[Title of the Invention]

METHOD FOR CONTROLLING CONTROL STATION CONFIGURATION IN MULTIPLE COMPUTER SYSTEM

Claim for the Patent

1. A method for controlling a control station configuration in a multiple computer system,

wherein the multiple computer system comprises:

a plurality of computers;

a common memory connected to the plurality of computers and commonly used among the computers;

a common bus connecting the plurality of computers;

a plurality of input/output units connected to the common bus, and controlled by the computers; and

a plurality of control stations connected to the common bus, and controlled so as to control and manage the common bus as a master station on its own at a time,

wherein each computer and the plurality of control stations are combined and connected, each control station reports status signal indicating whether or not the common bus is normally used to the combined computer, each computer refers to configuration control information stored in the common memory based on the status signal from the combined control station and an instruction from another computer, updates the configuration control information, and issues a configuration control instruction to the combined control station.

Detailed Description of the Invention

[Application Field of the Invention]

The present invention relates to such a multiple computer system that a plurality of computers control a plurality of input/output units connected to a common bus, and specifically, to a configuration control method for a plurality of control stations including a control function which controls and manages the common bus.

Here, "configuration control" is to control the control stations so that only one of the plurality of control stations connected to a common bus always controls and manages the common bus as a master station, and if the master station breaks down, one of remaining and waiting control stations (monitor stations) immediately replaces the previous master station and controls and manages the common bus.

Here, "control and manage" is to control and manage every thing for using the common bus, and in such a system that a computer occupies the common bus to control a specific input/output unit, it is an important job to occupy and control the common bus.

Hereinafter, a control station which controls and manages is referred to as master CST, a control station which may become a master CST instead of the master CST when the master CST breaks down is referred to as monitor CST, and other control stations are referred to as idle CST.

[Conventional Art]

One embodiment of a common bus is a loop-type transmission line.

In recent years, a loop-type transmission line which is multiplexed to improve the reliability of a transmission line itself has been proposed. In addition, a plurality of CSTs are connected to the loop-type transmission line, any one of the CSTs is designated as a master CST, and others are designated as a monitor CST or an idle CST, so that even if a specific CST breaks down, a system connected to the loop-type transmission line is caused not to break down as a whole system with the configuration control by CST.

Conventionally, it has been supposed that CSTs are connected to a common bus independently from a computer, and each CST originally determines a master CST and a monitor CST.

That is, it is supposed that the monitor CST monitors signal on the common bus to decide whether or not the master CST has

broken down, and if it is decided that the master CST has broken down, designates its own CST as a master CST to control and manage the common bus. In this case, if there are two or more monitor CSTs, while it is predetermined in many cases which one of the monitor CSTs becomes a master CST, such management is complex, and two monitor CSTs may try together to become a master CST.

And, even if the master CST itself has not broken down, if signal on the common bus has ceased, the monitor CST may decide that it is a failure of the master CST to try to designate itself as a master CST.

Therefore, conventionally, while it has been tried to prevent such disadvantage by providing specific signal line among CSTs to communicate signal, the secure configuration control for CSTs has been difficult because of the complexity of the control.

[Object of the Invention]

The purpose of the present invention is to provide a configuration control method for control station in a multiple computer system which securely controls so that, among a plurality of CSTs on the common bus, only one CST is designated as a master CST, and other CSTs are designated as a monitor CST or an idle CST.

[Summary of the Invention]

The features of the present invention are that a plurality of computers which control input/output units connected to the

common bus and each CST are combined and connected, and a monitor CST is designated as a master CST according to an instruction from the combined computer, that each CST reports the status of the CST itself and the status of signal on the common bus to the combined computer, and that such an information that which CST is a master CST is stored and updated in a common memory commonly used by each computer, and each computer refers to configuration control information of the common memory to perform configuration control according to the report from a combined CST and instructions from other computers.

According to the present invention, the configuration control information is managed on one common memory, so that two or more CSTs are not designated by mistake as a master CST.

When receiving a failure report of a CST itself from the master CST, a computer connected to the master CST refers to the configuration control information of the common memory, recognizes that a CST connected to itself is the master CST, immediately rewrites this CST to an idle CST, and reports to other computers that such CST has broken down.

A computer which has received this report refers to the configuration control information of the common memory, if a CST connected to itself is the monitor CST, rewrites this CST to a master CST, and instructs the CST connected to itself to become a master CST.

In such a case, the common memory can not be concurrently accessed from two computers, so that a secure configuration control can be performed.

[Embodiments of the Invention]

Fig. 1 is an exemplary embodiment of a multiple computer system to which the present invention is applied.

In Fig. 1, 1 to 3 are computers (hereinafter, referred to as CPU), internally include program, and execute this program to control a plurality of input/output units 16 to 18. 4 is a common memory commonly used by each CPU 1 to 3, and as described below, stores configuration control information of CSTs. 5 to 7 are stations (hereinafter, referred to as ST) connecting each CPU 1 to 3 to common buses, herein loop-type transmission lines 11 and 12 which are reversely directed each other, and 8 to 10 are control stations (CST) including control and manage function for the loop-type transmission lines 11 and 12, CST 8 to 10 are connected to combined CPU 1, 2, and 3 through connection lines 80, 90, and 100 respectively. 13 to 15 are micro stations (hereinafter, referred to as MST) connecting input/output units (hereinafter, referred to as I/O) 16 to 18 to the loop-type transmission lines 11 and 12. 19 is common signal line connecting each CPU 1, 2, and 3, and, as described below, is used for configuration control of a CST. Each MST includes a configuration for memorizing a CPU whose request should be served by I/O connected to itself, and a function for folding back a loop based on an instruction from a CST. A CPU whose request should be served is memorized when such MST receives an occupation request from the CPU in such a status (referred to as neutral status) that a CPU to be served is not fixed, and this occupation status is terminated when an instruction for canceling the occupation is received from the occupying CPU, and

when an instruction for canceling the occupation is forcibly received from other un-occupying CPU. This occupation request instruction is referred to as CONC. RSV (Connect & Reserve) command, a cancel instruction for an occupation status is referred to as CONC.FREE (Connect & Free) command, and an instruction for forcibly canceling an occupation is referred to as RESET.FREE (Reset & Free) command. When an instruction for an I/O connected under a MST is transferred from a CPU, if it is an instruction from other than CPU occupying the I/O, the MST does not receive the instruction as long as it is not a RESET.FREE command. By providing this mechanism, an I/O connected to a loop bus through a MST is appropriately shared by a plurality of CPUs. That is, even if an input/output request is issued by mistake from user program of the CPU 2 to the I/O 16 occupied from the CPU 1, the I/O 16 does not disturb a processing which is currently served. That is because the input/output request from the CPU 2 is rejected by the MST.

Each CPU executes such management of I/O connection status, and a table for managing this I/O connection status is placed in a memory of each CPU. When it becomes to be necessary to switch occupation status of an I/O, an operating system (OS) of each CPU communicates to a CPU currently occupying the I/O to request it to issue a CONC.FREE command to the I/O. The CPU which has received the request issues the CONC.FREE command to the I/O, cancels the occupation, and returns such a response that the occupation has been canceled to the CPU which requested the cancellation. The CPU which requested the cancellation receives this response, and issues a CONC.RSV command to occupy the I/O.

When it is detected that a CPU currently occupying the I/O is in a stopping status, or is not normally operating, the CPU which requested the cancellation issues a CONC.FREE command to cancel an occupation status of the I/O, and after that, issues a CONC.RSV command to occupy the I/O. According to such transition of the I/O occupation, while a management table in each CPU is rewritten, this management table may be placed in the common memory 4 among CPUs.

A plurality of CSTs are connected to a same loop to increase the reliability of a system. This is because that if the number of CSTs is one, when the CST may break down, or stations of both sides of the CST become to be abnormal, control information becomes to be unable to be transmitted from the CST to other stations, and the loop becomes to be unable to function. If such a plurality of CSTs are caused to control independent loops each other, the plurality of CSTs may execute conflicting control each other, so that only one CST of the plurality of CSTs is provided with the authority for controlling a loop. The CST provided with this authority is referred to as a Master mode a CST (master CST). Other CSTs are in a mode (Moniter mode) of monitoring the status of a loop, or in a mode (Idle mode) of operating just passively, and a CST in a Monitor mode is referred to as a monitor CST. Only the master CST can instruct MST and ST to fold back a loop, and also, can instruct to switching the loop 11 usually used for data transfer between a CPU and an I/O, and the loop 12 used for transferring monitor signal of a loop, etc. The master CST transfers monitor signal to a loop. The monitor CST monitors the monitor signal

transferred from the master CST, and if this monitor signal can not be detected during a certain term, decides that the master CST has broken down, interrupts to a CPU, and reports the abnormality of the master CST. If a signal disconnection of a loop is detected, the master CST tries to switch the duplicated loop (switching of the loop 11 and the loop 12). If the operation can be continued by this switching, nothing else is required to do; otherwise, a loop back operation (folding back a loop) is instructed to each station based on configuration information (indicating which station is connected with what relative relation (topology)) of the loop memorized in its own CST. When a CST is instructed from a CPU to become a master, the configuration information of a loop is transferred from the CPU to the CST, however, before it is instructed to become a master, the configuration information may be previously transferred to each CST. When signal disconnection is still detected by the master CST even if the master CST tries such loop control, and also, when the monitor CST detects its own abnormality, the master CST interrupts to a CPU to move from the master CST to an idle CST. Idle CSTs do not actively work for a loop, and also, do not monitor the status of a loop. If the idle CST or the monitor CST is instructed to become a master (this instruction is referred to as a master command), it becomes a master to obtain the authority for controlling a loop. The master CST moves to a monitor CST according to a reset instruction (rest command) from a CPU. Such transition of the status is illustrated in Fig. 2. If the electric power is applied to a CST, it becomes to be in an Idle mode, and if it receives a CONC.RSV

command from a CPU, it moves to a Monitor mode. This is because when a system is caused to initially start, unnecessary error processing because of the difference of the electric power on sequence is prevented. That is, if the electric power is initially applied to a CST, and the CST is caused to be a monitor CST, the monitor CST starts monitoring a loop. While a Monitor mode CST may decide that a loop is abnormal (Master mode CST is abnormal) until a Master mode CST is defined after such timing of the monitoring start (because a Master mode CST is defined by a instruction from a CPU, the Master mode CST does not exist until the CPU becomes operable after the electric power has been applied to the CPU), if a CST moves to a Monitor mode according to an instruction from a CPU, such a problem is not induced.

Fig. 3 illustrates a transition of operation modes of a loop bus. NL operation is a normal operation embodiment, transfers data between a CPU and an I/O with one loop (this is referred to as NL: Normal Loop) of duplicated loops, and monitors the loops with the other loop (this is referred to as BL). If the abnormality is detected in the Normal Loop during the NL operation, data is transferred between a CPU and an I/O in the Back Loop. If both of the Normal Loop and the Back Loop become abnormal (for example, two lines of the loops are broken), and loop rounding signal becomes to be undetectable, the master CST instructs each station to fold back a loop, and instructs configuration control (loop back control) of the loop so that meaningful signal can be transferred rounding a loop (loop which combines Normal loop, its folding back point, Back loop, and its

folding back point). Thereby, An operation mode is referred to as LB operation (Loop Back operation), which includes meaningful signal rounding a loop, and functions as a loop bus, and a station sandwiched by folding back points becomes to be in such a status that it is disconnected from the loop. In Fig. 3, a dash line arrow indicates that when a command for return to normal operation (RLBC command) is issued from a CPU to a CST in a Master mode, an operation mode moves. A CPU can instruct a Master mode CST to move to a LB operation (SLBC command). In this case, while a station to be a folding back point is indicated to the Master mode CST, the Master mode CST processes according to memorized configuration information of a loop.

The above operations will be actually described using the drawings.

Here, the CPU 1 and the CST 8 connected to the CPU 1 will be described as an example. The CPU 2, 3 and the CST 9, and 10 are also same.

Fig. 4 illustrates a block diagram of the CST 8. The CST 8 includes a microprocessor 8-1, a read only memory (ROM) 8-2 internally including micro program, a writable memory (RAM) 8-3, an interrupt control LSI (PICU) 8-4, a communication control LSI (HDLC) 8-5, a general purpose input/output control LSI (PPI) 8-6, a timer (PTM) 8-7, and an interrupt register 8-9 memorizing content of interrupt from the CPU 1, and such components are connected with a common bus 8-8.

Further, the CST 8 includes a multiplexer (MPXB) 8-12 selecting one of a monitor signal from a monitor signal generator 8-11, which prompts data transfer, and data signal

from the HDLC 8-5, a multiplexer (MPXA) 8-17 selecting one to be transmitted of signal selected by the multiplexer (MPXB) 8-12, and direct signal on a loop, and a reverse directional multiplexer (MPXD) 8-22, a detector 8-15 detecting signal break of all loops, a Loop Control 8-14 issuing a selection instruction to such multiplexers, signal receiving circuits 8-20 and 21, and signal transfer circuits 8-18 and 19. And, a report register 8-27 is used for reporting to a connected computer. The interrupt register 8-9 and the report register 8-27 are connected to the CPU 1 through a connection line 80. In the above configuration, an actual processing will be described by using Fig. 5 to Fig. 9. Fig. 5 illustrates a processing initiated by an instruction from the CPU 1 in the CST 8, or electric power on of the CST 8. In case of the electric power on, the CST 8, first, unconditionally moves to an IDLE mode, and also memorizes the IDLE mode in the RAM 8-3. In an IDLE mode, the CST 8 itself moves to a Pass status for the Loop 11 and 12. This is performed by the Loop Control 8-14 and the multiplexer 8-17 or 8-22. If an instruction from the CPU 1 is a MASTER instruction, the CST 8 memorizes in the RAM 8-3 that the CST 8 itself is in a MASTER mode, transfers a monitor signal to the Loop 11 (Normal Loop), and also initiates a monitor signal rounding wait timer 8-7. If an instruction from the CPU 1 is reset (Reset), the CST 8 itself moves to a MONITOR mode, and moves to a PASS status for the Loop 11 and 12. Further, the monitor signal wait timer 8-7 from other MASTER CST is initiated. Fig. 6 illustrates an example of interrupt processing initiated by an interrupt from a currently used loop. Here, [currently

used loop] is the Loop 11 while Normal Loop is operating, or the Loop 12 while Back Loop is operating. The HDLC 8-5 receives an interrupt from the Loop 11 or 12 through the multiplexer 8-13, and the interrupt from the HDLC 8-5 is inputted by the interrupt control LSI 8-4 to the MPU 8-1. An interrupt processing program illustrated in Fig. 6 is initiated by this interrupt. At this time, if the received data is monitor signal, the MONITOR CST reinitiates the monitor signal wait timer 8-7. And, this interrupt when HDLC LSI is in a failure status, if the CST 8 itself is the MASTER CST, it decides that it is unable to continue to manage the Loop 11 and 12, moves to Pass status for the Loops, and moves to an IDLE mode. Further, the CST 8 sets the MASTER failure of its own CST 8 to the report register 8-27 and reports to the connected CPU.

Fig. 7 is an example of a processing for checking signal break of the currently used Loop 11 or 12. The checking of signal break is executed only by the MASTER CST, the signal break is detected by the signal break detector 8-15, and is detected by reflecting to the status of the general purpose input/output control LSI. If the signal break is induced while the Normal Loop (Loop 11) is operating, the currently used loop is switched to the Back Loop (Loop 12). The switching processing cuts off the multiplexer A 8-17 according to an instruction of the Loop CTL 8-14, and transfers signal from the multiplexer B 8-12 to the Loop 12 with the multiplexer D 8-22. Further, a monitor signal is transferred to the Back Loop (BL) 12, and also a Normal Loop failure is reported to the CPU 1. When signal is broken, if the Back Loop is operating, a Loop Back instruction

is issued to the ST 5, 6, and 7, and MST 13, 14, and 15 which are connected to the Loop 11 and 12, it moves to the Loop Back mode. And, the abnormality of the Back Loop is reported to the CPU 1. If signal is broken while the Loop Back is operating, it is decided that the Loop 11 and 12 can not be managed, and its own MASTER CST Give up is reported to the connected CPU 1. Fig. 8 illustrates an example of a time out processing in a CST, when a CST mode memorized in the RAM 8-3 is a MONITOR mode, if POL signal from other MASTER mode CST is not received within a predetermined time, a real time out is induced. If the real time out is induced, it is decided that other MASTER mode CST has failed, and that is reported to the CPU 1 with an interrupt. If a CST mode memorized in the RAM 8-3 is a MASTER mode, a monitor signal transferred by its own CST 8 becomes to be in monitor signal rounding wait time out within a certain time, and if it is a real time out, it reports its own MASTER CST failure to the CPU 1 to moves to an IDLE mode. Fig. 9 is an example of a processing of such a case that communication between its own CST and a CPU is ceased. In this example, when a report to the CPU 1 is executed with DMA (Direct Memory Access), if DMA error is induced, it is estimated that communication between its own CST and the connected CPU is ceased, and if a CST mode in the RAM 8-3 is a MASTER mode, it moves to an IDLE mode, and causes the Loop to be in a PASS status.

While I/Os (16 to 18) connected to a loop can be occupied by different CPUs respectively, it is also possible to cause all I/Os connected to a loop to be occupied as a whole by a CPU. To do this, the CPU occupying a loop as a whole issues a CONC.FREE

command to all I/Os under the loop, and a CPU which wants to newly occupy the loop issues a CONC.RSV command to all I/Os under the loop. In this case, if the loop is in a status of LB operation, and an I/O is disconnected from the loop, a CONC.RSV command does not reach the I/O, so that only the I/O is not occupied by the CPU which has newly occupied the loop. To prevent this disadvantage, a CPU memorizes the disconnected I/O which is reported by an interrupt from a CST in a MASTER mode, and after the loop is returned to a normal operation (after a RLBC command is issued to a CST in a MASTER mode), a CONC.RSV command is issued to the I/O. Thereby, regardless of the operation status of a loop, it becomes to be able to occupy a loop as a whole and switch the occupation among each CPU.

By the way, in a multiple computer system, the configuration control of CPU is executed. For example, in Fig. 1, it is assumed that operations A, B, and C are being executed by the CPUs 1, 2 and 3 respectively, and the operation A is an important job for the whole system, when the CPU 1 becomes abnormal to break down, even if the remaining CPU 2 and 3 continue the operations B and C, it may be meaningless as the whole system. Thus, in such a case, the operation A which has been being executed by the CPU 1 must be taken over by the CPU 2 or 3. In addition, it is necessary to release all resources which the abnormal CPU 1 has been occupying, and to cause the CPU 1 to cease so as not to induce failure and disturbance for other CPU 2 and 3 to process. Such a processing is referred to as a configuration control of a CPU. Even if a CPU is not abnormal, for convenience of system operation, operation of some

CPU may be caused to stop, and operation content executed in a CPU may be switched according to a plan. This is also referred to as a configuration control of a CPU. When such a configuration control is executed, and when such a configuration control is executed that a CPU to which a MASTER mode CST is connected ceases as a result, it is necessary to cause a CST connected to a CPU connected to another surviving CPU to be in a MASTER mode. This is because a MASTER mode CST can not report the abnormality of a loop if the CPU connected to it is stopping, and also the memorized configuration information of a loop can not be updated according to an instruction from the CPU.

A CPU which receives from a MASTER mode CST such a communication that the CST has become to be unable to keep a MASTER mode communicates to one of other CPUs to cause a connected CST to be in a MASTER mode.

The switching of a MASTER mode CST by a configuration control of a CPU is executed as illustrated in Fig. 10. In Fig. 10, [Disconnect CPU] is such a processing that a target CPU is caused to stop, and the resources which the CPU has been occupying is released. [Switch CPU] is such a processing that operation which the CPU has been executing until then is caused to stop, the resources which the operation has been occupying is caused to be released, and a new operation is started. [Start up CPU] is such a processing that a CPU which has been stopping until then is caused to start its operation, and a new operation is started. The processing flows of [Disconnect CPU] and [Start up CPU] are illustrated in Fig. 11 and 12 respectively. In Fig. 12, it is possible to check whether or not a loop is connected

to a CPU, for example, by having a table as illustrated in Fig. 13. In Fig. 13, 31 is stored in a common memory CM (4 in Fig. 1), a common memory among CPUs, which can be referred from each CPU. The number of the tables is same as the number of loop buses of a whole system (loop 1 to loop n), and each bit 0, 1, 2, … is caused to correspond to the number of CPU, that is, it is indicated that if bit 2 for the loop 2 is "1", the loop 2 is connected to a CPU whose CPU# is 2, and if it is "0", the loop 2 is not connected to a CPU whose CPU# is 2. In addition, this table is also used to request to cause a CST under another CPU to be in a MASTER mode. A table whose structure is same as that of Fig. 13 is also used to determine which one of CSTs connected to a loop is in a MASTER mode. However, when a MASTER mode CST is managed, if the MASTER mode CST is indicated by "1", in Fig. 13, a bit which is "1" in a table for each loop is only one in each line, and this is different from such a case that a connection relation between each loop and a CPU is indicated. Fig. 14 indicates whether or not a MASTER mode CST is being set to each loop, and is used to interlock so that two CSTs are not set to a MASTER mode at same time. Meanwhile, when such tables are referred to, because it is necessary to interlock so that it is not referred to from a plurality of CPUs at same time, if this table is placed in the CM 4 which is a common memory among CPUs, it is interlocked by using a TEST and SET instruction, etc.

Next, a configuration control of CSTs by a report from the CST side is illustrated in Fig. 15 and Fig. 16. In Fig. 11, Fig. 12, and Fig. 16, when a communication is issued so as to set a MASTER mode to another CPU, a method for determining a

communication partner can be executed, for example, by searching CSTs connected to a loop from loops of Fig. 13 and connection information of CSTs, and selecting any one from CPUs connecting the CSTs excluding CPUs to be disconnected and CPUs which are currently stopping. This is executed by using the common signal line 19 of Fig. 1. Meanwhile, while this communication may be issued after one CPU has been selected as described above, the communication is issued to all CPUs, then, based on first come, first served, a CPU which receives the communication may also cause a CST connected to the CPU to be in a MASTER mode. However, in such a case, a CPU currently connecting the CST which has been in a MASTER mode must ignore this communication.

[Advantage of the Invention]

As described above, according to the present invention, each computer manages the status of all CST as a whole by using a common memory, and executes configuration control of each CST, so that competition for mode of each CST, for example, such a disadvantage is eliminated that a plurality of CSTs managing a common bus as a whole are induced, and the common bus becomes to be unable to be managed, so that the reliability of a multiple computer system can be improved by executing secure configuration control of CST.

Brief Description of the Drawings

Fig. 1 is a diagram illustrating an exemplary embodiment of a multiple computer system to which the present invention is applied; Fig. 2 is a diagram illustrating a status of transition

of mode of CST; Fig. 3 is a diagram illustrating a transition of operation mode of loop bus; Fig. 4 is a block diagram illustrating an exemplary embodiment of CST used for the present invention; Figs. 5 to 9 are flowcharts used for operation description of Fig. 4 respectively; Fig. 10 is a diagram illustrating a switching control of a MASTER mode CST by configuration control of CPU; Fig. 11 is a flowchart illustrating a processing of an instruction for disconnecting a CPU; Fig. 12 is a flowchart illustrating a processing for starting up a CPU; Fig. 13 is a diagram illustrating configuration control information utilized for operation description of Fig. 12; Fig. 14 is a diagram illustrating an example of configuration control information used while a MASTER mode CST is being set; Fig. 15 is a flowchart illustrating an interrupt processing of CPU connecting a MASTER mode CST; and Fig. 16 is a flowchart illustrating an interrupt processing of CPU connecting a CST which is not in a MASTER mode.

1 to 3 computer;
4 common memory;
8 to 10 control station (CST);
11, 12 loop-type transmission line (common bus);
80, 90, 100 connection line.

Agent Patent Attorney Akio TAKAHASHI

ALP4490/ALP4491

Fig. 1

4 COMMON MEMORY

Fig. 2

#1 MASTER MODE

#2 MONITOR MODE

#3 IDLE MODE

#4 ABNORMALITY DETECTION

#5 MASTER COMMAND FROM CPU

#6 RESET COMMAND FROM CPU

#7 CONC.RSV COMMAND FROM CPU

Fig. 3

#1 NL OPERATION

#2 BL OPERATION

#3 LB OPERATION

Fig. 4

#1 TO CPU1

Fig. 5

#1 CST INITIATION BY CPU 1

#2 MASTER MODE INSTRUCTION?

#3 MEMORIZE ITS OWN CST MASTER MODE

#4 MEMORIZE NORMAL OPERATION MODE

#5 TRANSFER MONITOR SIGNAL TO LOOP 11

#6 INITIATE MONITOR SIGNAL ROUNDING WAIT TIMER

#7 ITS OWN CST RESET INSTRUCTION?

#8 MEMORIZE ITS OWN CST IDLE MODE

#9 INITIATE MONITOR SIGNAL WAIT TIMER FROM OTHER CST MASTER

#10 MEMORIZE ITS OWN CST MONITOR MODE

#11 SET ITS OWN CST PASS STATUS

Fig. 6

#1 MONITOR SIGNAL RECEIVING?

#2 HDLC ERROR?

#3 ITS OWN CST IS IN MASTER MODE?

#4 MOVE ITS OWN CST TO PASS STATUS FOR LOOP AND MEMORIZE IDLE MODE

#5 ISSUE ATTENTION TO CPU1 (REPORT ITS OWN CST MASTER FAILURE)

#6 RESTART MONITOR SIGNAL WAIT TIMER

Fig. 7

#1 ITS OWN CST IS IN MASTER MODE?

#2 SIGNAL BREAK?

#3 NORMAL OPERATION MODE?

#4 MEMORIZE B.L OPERATION MODE

#5 TRANSFER MONITOR SIGNAL TO B.L

#6 REPORT N.L ABNORMALITY TO CPU 1

#7 BACK LOOP OPERATION MODE?

#8 MEMORIZE L.B OPERATION MODE

#9 ISSUE L.B INSTRUCTION TO EACH ST

#10 REPORT B.L ABNORMALITY TO CPU 1

#11 REPORT ITS OWN MASTER CST GIVE UP TO CPU1

Fig. 8

ALP4490/ALP4491

#1 ITS OWN CST IS IN MONITOR MODE?

#2 MONITOR SIGNAL WAIT TIME OUT?

#3 REPORT OTHER MASTER MODE CST FAILURE TO CPU 1 WITH INTERRUPT

#4 ITS OWN CST IS IN MASTER MODE?

#5 MONITOR SIGNAL ROUNDING WAIT TIME OUT?

#6 TIME OUT RETRY OVER?

#7 REPORT ITS OWN MASTER CST FAILURE TO CPU 1 WITH INTERRUPT AND MOVE TO IDLE MODE

#8 TRANSFER MONITOR SIGNAL

#9 RETRY COUNTER +1

Fig. 9

#1 SET REPORT CONTENT TO REPORT REGISTER 8 TO 27 AND REPORT

#2 NORMAL REPORT TERMINATION?

#3 MOVE ITS OWN CST TO PASS STATUS FOR LOOP AND MEMORIZE IDLE MODE

Fig. 10

#1 CONFIGURATION CONTROL OF CPU

#2 DISCONNECT

#3 SWITCH

#4 START UP

#5 CONDITION FOR SETTING MASTER MODE CST

#6 WHEN MASTER MODE CST IS CONNECTED TO CPU TO BE DISCONNECTED

#7 MASTER MODE CST IS NOT CHANGED

#8 WHEN MASTER MODE CST DOES NOT EXIST IN LOOP CONNECTED TO CPU EXECUTING START UP PROCESSING

#9 CST NEWLY BECOMING MASTER MODE

#10 CST CONNECTED TO CPU EXECUTING DISCONNECT PROCESSING

#11 CST CONNECTED TO CPU EXECUTING START UP PROCESSING

Fig. 11

#1 INSTRUCTION FOR DISCONNECTING CPU

#2 MASTER MODE CST IS CONNECTED TO CPU TO BE DISCONNECTED?

#3 CANCEL MASTER REGISTRATION OF THE CST

#4 LOOP THE CST IS CONNECTED TO IS ALSO CONNECTED TO ITS OWN CPU?

#5 RESET THE CST

#6 REGISTER CST CONNECTED TO ITS OWN CPU OF THE LOOP AS MASTER, AND SET INDICATION THAT MASTER MODE IS BEING SET TO ISSUE MASTER COMMAND

#7 MASTER COMMAND HAS BEEN NORMALLY PROCESSED?

#8 CLEAR INDICATION THAT MASTER MODE IS BEING SET

#9 ALL LOOPS HAVE BEEN CHECKED?

#10 CANCEL MASTER REGISTRATION OF CST REGISTERED AS MASTER OF THE LOOP, CLEAR INDICATION THAT MASTER MODE IS BEING SET, AND COMMUNICATE TO ANOTHER CPU TO SET MASTER MODE CST

Fig. 12

#1 START UP CPU

#2 MASTER MODE CST EXISTS IN LOOP CONNECTED TO ITS OWN CPU?

#3 THE LOOP IS CONNECTED TO CST OF ITS OWN CPU?

#4 SET INDICATION THAT MASTER MODE IS BEING SET

#5 REGISTER ITS OWN CST AS MASTER TO ISSUE MASTER COMMAND

#6 MASTER COMMAND HAS BEEN NORMALLY PROCESSED?

#7 CLEAR INDICATION THAT MASTER MODE IS BEING SET

#8 ALL LOOPS CONNECTED TO ITS OWN CPU HAVE BEEN CHECKED?

#9 CANCEL MASTER REGISTRATION OF THE CST, CLEAR INDICATION THAT MASTER MODE IS BEING SET, AND REQUEST ANOTHER CPU TO SET MASTER MODE

Fig. 13

#1 FOR LOOP 1

#2 FOR LOOP 2

#3 FOR LOOP n-1

#4 FOR LOOP n

Fig. 14

#1 FOR LOOP 1

#2 FOR LOOP n-1

#3 FOR LOOP n

Fig. 15

#1 MASTER MODE MAINTENANCE IMPOSSIBILITY INTERRUPT

#2 ANOTHER CST IS CONNECTED TO LOOP THE CST IS CONNECTED TO?

#3 CST WHICH HAS BEEN CONNECTED TO ANOTHER IS CONNECTED TO CPU TO BE DISCONNECTED?

#4 COMMUNICATE THE CPU TO CAUSE THE CST TO BE IN MASTER MODE

#5 ALL CST CONNECTED TO THE LOOP HAVE BEEN CHECKED?

Fig. 16

#1 MASTER MODE SETTING REQUEST COMMUNICATION FROM ANOTHER CPU

#2 ABNORMALITY DETECTION INTERRUPT OF MASTER MODE CST

#3 MASTER MODE IS BEING SET?

#4 SET INDICATION THAT MASTER MODE IS BEING SET

#5 CONNECT TO COMMUNICATED LOOP, REGISTER CST CONNECTED TO ITS OWN CPU TO MASTER, AND ISSUE MASTER COMMAND

#6 MASTER COMMAND HAS BEEN NORMALLY PROCESSED?

#7 CLEAR INDICATION THAT MASTER MODE IS BEING SET

#8 CLEAR INDICATION THAT MASTER MODE IS BEING SET, CANCEL MASTER REGISTRATION, AND REQUEST ANOTHER CPU TO SET MASTER

とするようにしてもよい。但しその場合、現にMasterモードであつたＣＳＴを接続しているＣＰＵはこの連絡を無視しなくてはいけない。

〔発明の効果〕

このように、本発明によれば、全ＣＳＴの状態を共通メモリを用いて各計算機が一括管理して、各ＣＳＴの構成制御を行なうので、各ＣＳＴのモードの競合、例えば共通バスを一括管理するCSTが複数個発生し、共通バスの管理不能状態を招くという不都合がなくなり、確実なＣＳＴの構成制御を行うことによりマルチ計算機システムの信頼性を向上できる。

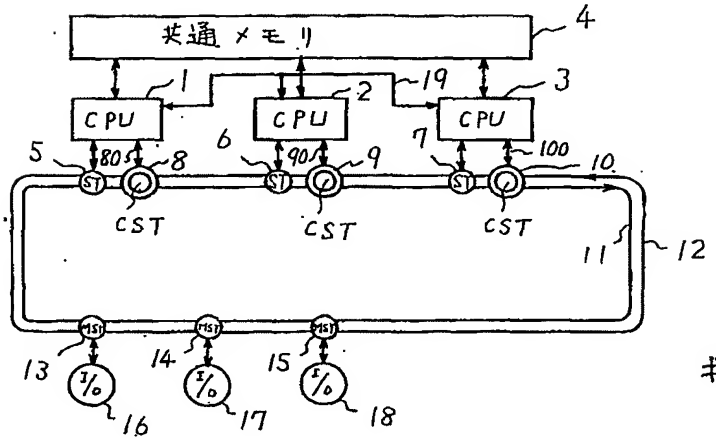
図面の簡単な説明

第１図は本発明が適用されるマルチ計算機システムの一実施例構成図、第２図はＣＳＴのモードの状態遷移を示す図、第３図はループバスの運転モードの遷移を示す図、第４図は本発明に用いられるＣＳＴの一実施例ブロック図、第５図～第９図はそれぞれ第４図の動作説明に用いられるフローチャート、第１０図はＣＰＵの構成制御に伴うMasterモードのＣＳＴの切り換え制御を示す図、第１１図はＣＰＵの切り離し指令時の処理を示すフローチャート、第１２図はＣＰＵの立上げ時の処理を示すフローチャート、第１３図は第１２図の動作説明に利用される構成制御情報を示す図、第１４図はMasterモードのＣＳＴを設定処理中に用いられる構成制御情報の一例を示す図、第１５図はMasterモードのＣＳＴを接続しているＣＰＵの割込み処理を示すフローチャート、第１６図はMasterモードでないＣＳＴを接続しているＣＰＵの割込み処理を示すフローチャートである。

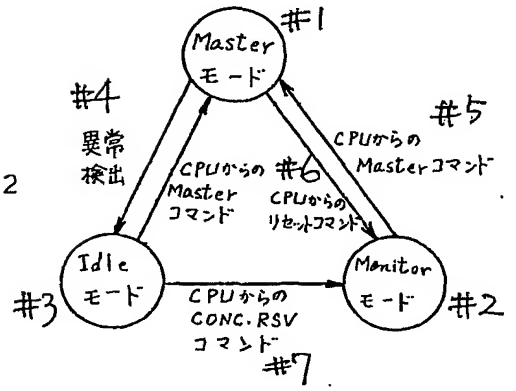
１～３…計算機、４…共通メモリ、８～１０…コントロールステーション（ＣＳＴ）、１１，１２…ループ状伝送路（共通バス）、８０，９０，１００…接続線。

代理人　弁理士　高橋明夫

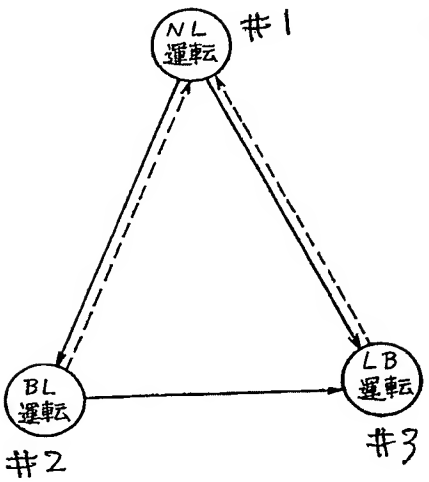
第１図（Fig.1）



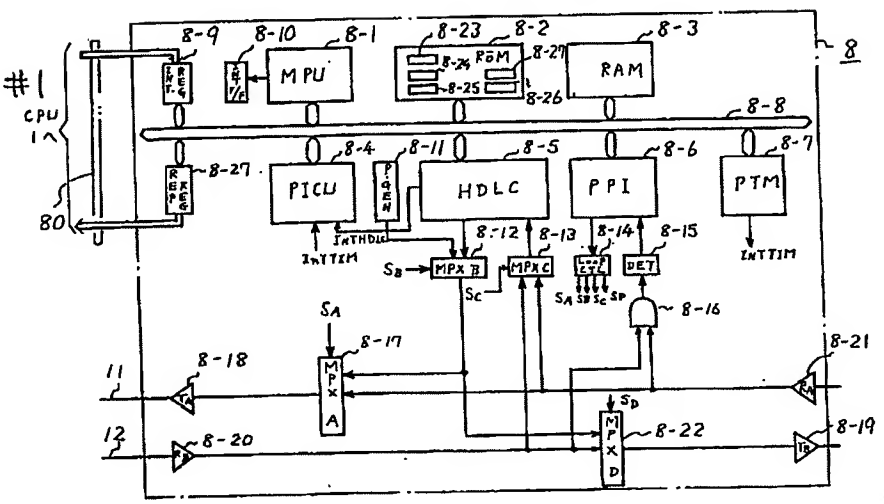
第２図（Fig.2）



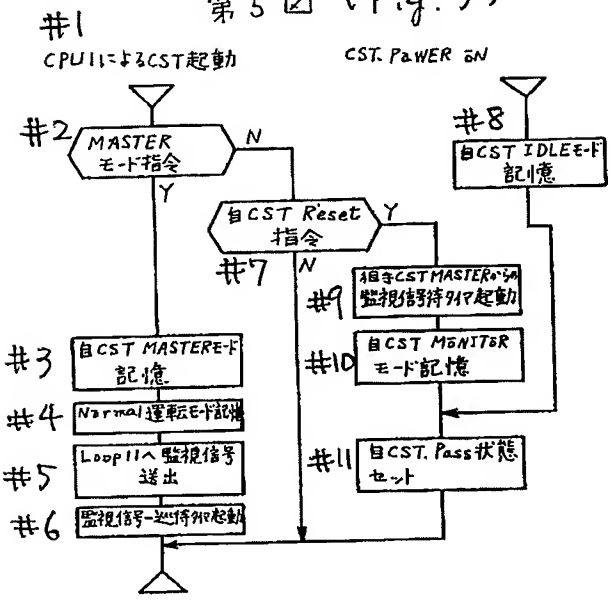
第3図 (Fig. 3)



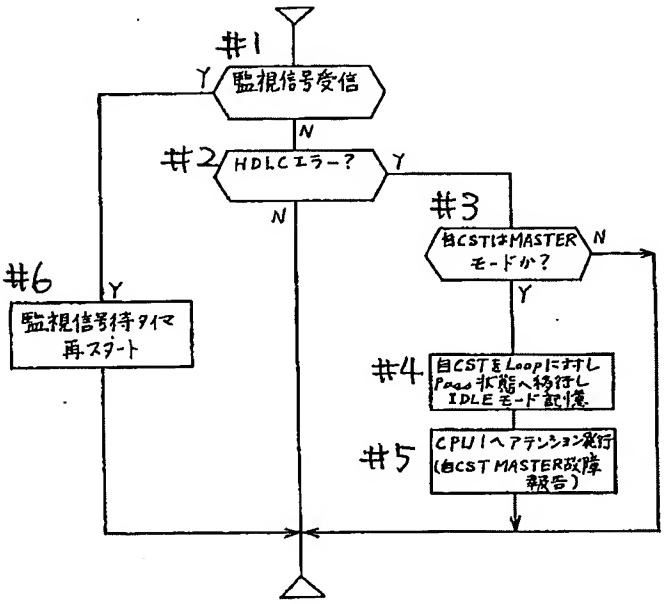
第4図 (Fig. 4)



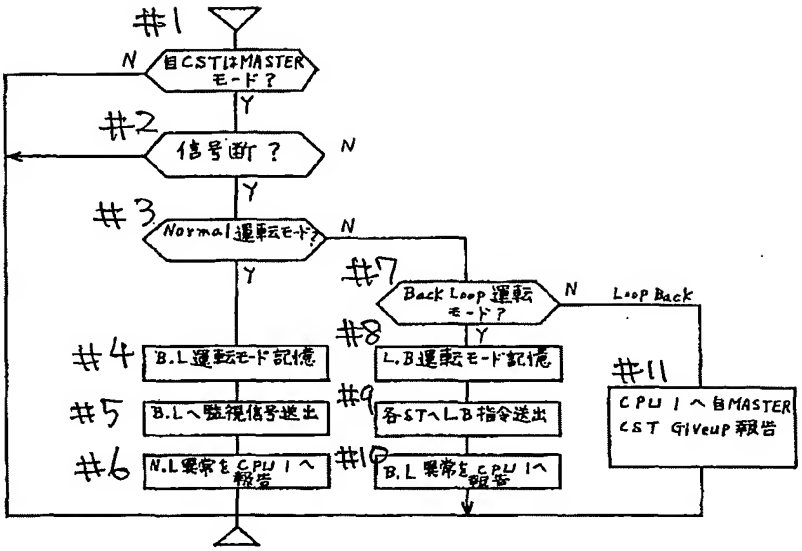
第5図 (Fig. 5)



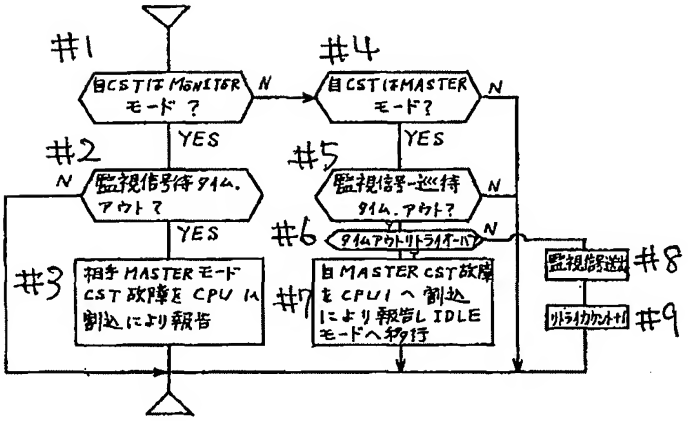
第6図 (Fig. 6)



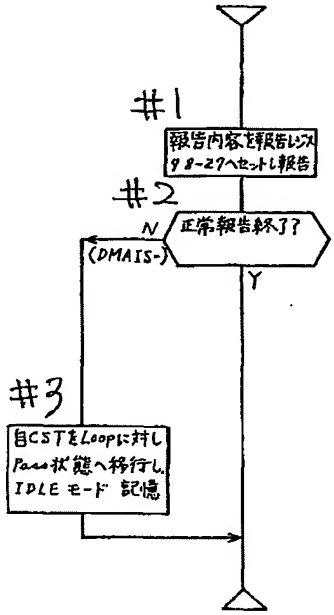
第 7 図 (Fig. 7)



第 8 図 (Fig. 8)



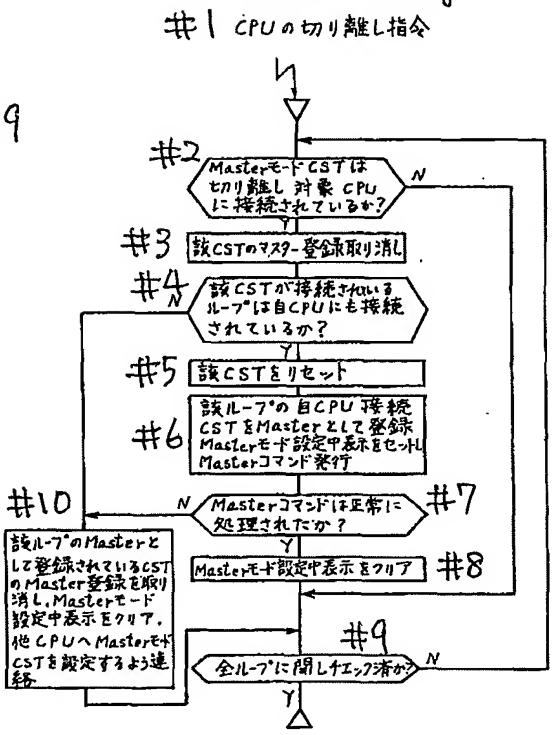
第 9 図 (Fig. 9)



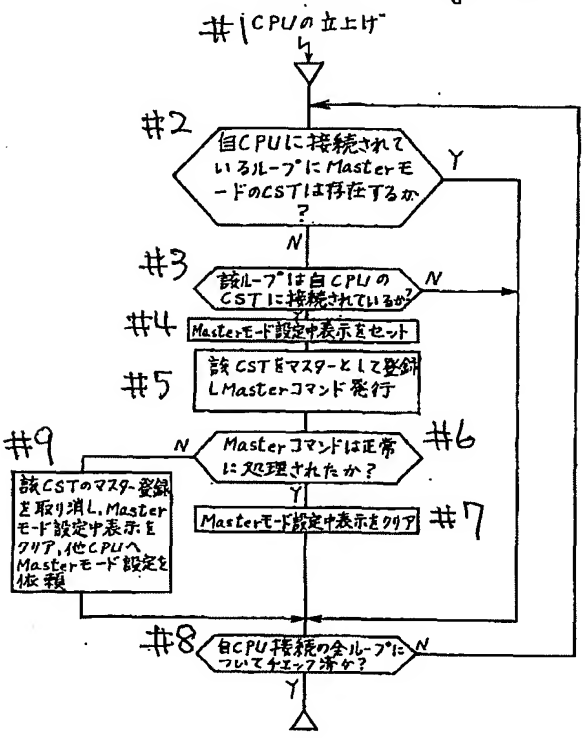
第10図 (Fig.10)

#1 CPUの構成制御	#5 Masterモード CST設定の条件	#9 新しくMasterモードとなるCST
切り離し #2	Masterモードの CSTが切り離し対象CPUに接続されているとき。 #6	切り離し処理を行なうCPUに接続されてるCST. #10
切り換え #3	Masterモードの CSTの変更はしない。 #7	—
立上げ #4	立上げ処理を行なうCPUに接続されているループにMasterモードのCSTが存在しないとき。 #8	立上げ処理を行なうCPUに接続されているCST #11

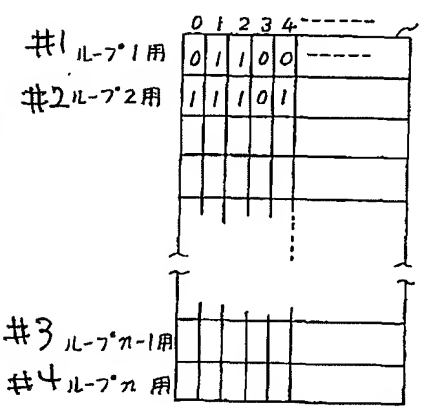
第11図 (Fig.11)



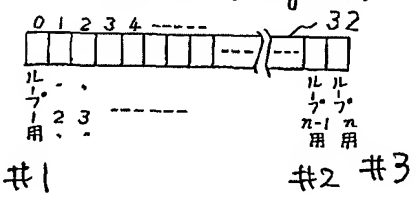
第12図 (Fig.12)



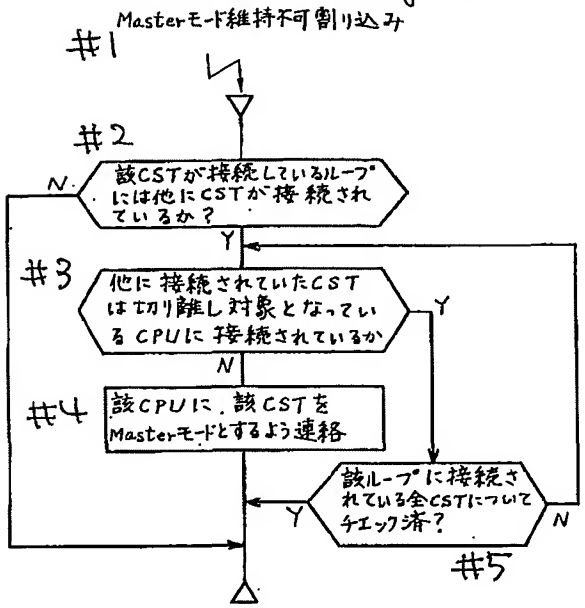
第13図 (Fig.13)



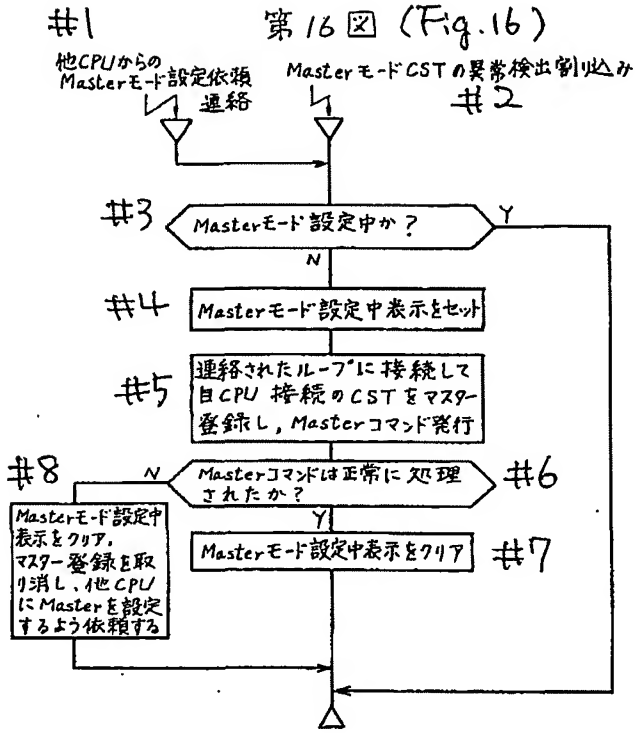
第14図 (Fig.14)



第15図 (Fig.15)



第16図 (Fig.16)



第1頁の続き

⑫発 明 者 岡田政和
日立市大みか町5丁目2番1号
株式会社日立製作所大みか工場内

⑫発 明 者 末木雅夫
日立市大みか町5丁目2番1号
株式会社日立製作所大みか工場内

⑫発 明 者 林慶治郎
日立市大みか町5丁目2番1号
株式会社日立製作所大みか工場内

⑫発 明 者 大貫健
日立市大みか町5丁目2番1号
株式会社日立製作所大みか工場内

⑫発 明 者 井手寿之
日立市大みか町5丁目2番1号
株式会社日立製作所大みか工場内

⑫発 明 者 溝河貞生
日立市大みか町5丁目2番1号
株式会社日立製作所大みか工場内